2009-12-08

EgoViz – a Mobile Based Spatial Interaction System

Keith Gardiner
Technological University Dublin, keith.gardiner@tudublin.ie

Junjun Yin
Technological University Dublin, junjun.yin@tudublin.ie

James Carswell
Technological University Dublin, jcarswell@tudublin.ie

Recommended Citation

Gardiner, K. Yin, J. & Carswell, J. (2009) EgoViz –A Mobile Based Spatial Interaction System. *Springer Verlag Lecture Notes in Computer Science;* Web & Wireless Geographic Information Systems (W2GIS'09); Maynooth Ireland, Dec. 7-8. doi:10.1007/978-3-642-10601-9_10

EgoViz –A Mobile Based Spatial Interaction System

K. Gardiner*, J. Yin, J.D. Carswell

Digital Media Centre,
Dublin Institute of Technology, Ireland
{keith.gardiner, junjun.yin, jcarswell}@dit.ie

Abstract. This paper describes research carried out in the area of mobile spatial interaction and the development of a mobile (i.e. on-device) version of a simulated web-based 2D directional query processor. The *TellMe* application integrates location (from GPS, GSM, WiFi) and orientation (from digital compass/tilt sensors) sensing technologies into an enhanced spatial query processing module capable of exploiting a mobile device's position and orientation for querying real-world 3D spatial datasets. This paper outlines the technique used to combine these technologies and the architecture needed to deploy them on a sensor enabled smartphone (i.e. Nokia 6210 Navigator). With all these sensor technologies now available on one device, it is possible to employ a personal query system that can work effectively in any environment using location and orientation as primary parameters for directional queries. In doing so, novel approaches for determining a user's query space in 3 dimensions based on line-of-sight and 3D visibility (ego-visibility) are also investigated. The result is a mobile application that is location, direction and orientation aware and using these data is able to identify objects (e.g. buildings, points-of-interest, etc.) by pointing at them or when they are in a specified field-of-view.

Keywords. MSI, Directional Query, Isovist, Radial Query

1 Introduction

This paper focuses on mobile directional querying and the development of algorithms for 2D and 3D spatial data processing. In conjunction with location, an orientation module provides angular data to a spatial query processor making it possible to perform directional queries from a mobile device in a real world environment.

The fundamental requirements for this type of service interaction are location, direction and orientation. Some typical applications in the current literature enable a user to point a mobile device at a building and, using position and direction, determine the building's address/identity (Simon and Frohlich, 2007b). This requires both accurate location and orientation as part of the query and the modules that provide this data in our case are the *LocateMe* and *DirectMe* modules. The locator component or *LocateMe* module is by design an open source, network independent mobile location determination application that can utilise GPS, Wi-Fi and GSM beacon information or any combination of them, to trilaterate location estimates (Kilfeather et al., 2007; Rooney et al., 2007a). The orientation component or

DirectMe module and a primary focus of this paper uses data from a number of sensors including a GPS sensor, a magnetometer sensor (digital compass) and an accelerometer sensor (tilt sensor). Using these data, the *TellMe* application formulates directional queries that are performed by a spatial database to determine if any spatial interaction exists between the query "window" and any of the buildings in our 3D university campus model. The shape of this query window takes on a variety of 2D and 3D forms from a simple ray to a polygon to a volume (see Figure 14). The results of this interaction are subsequently presented to the user in the form of a building address/details plus web-links to more information (e.g. classroom timetables, lab opening hours, etc.).

Primarily, the *TellMe* application provides a framework for processing 2-dimensional queries in an open, non-directional query-space (e.g. range query). This subsequently enables us to investigate some of the major issues in the area of mobile spatial interaction within 3D environments (e.g. 3D Visibility).

In relation to sensor data quality (e.g. noise), tests to compare the data gathered from a higher quality (i.e. more stable) external sensor packs with that of the integrated sensors on our mobile device are ongoing and will help to determine the suitability of current mobile devices for exploitation in the area of mobile spatial interaction. Following this, our attention shifts to exploring 3-dimensional visibility with the development of an *"Ego-Visibility"* query processor that further confines the query-space to simulate a user's view frustum. Results from this will enable us to investigate the possibility of providing full *"Hidden Query Removal"* functionality where only what the user can actually physically see gets returned by the directional query processor.

The remainder of the paper is organised as follows: Section 2 describes some related work in the area of directional querying. Section 3 outlines some methods used by current GIS to perform these queries using visibility analysis. Section 4 describes the design of the *TellMe* system which includes the *LocateMe* and *DirectMe* modules, the *TellMe Server* and hardware considerations. Section 5 details our approach to egocentric visibility and describes five different types of queries and Section 6 concludes with a summary and future work.

2 Related work

There have been a number of different applications proposed recently that utilise compasses and tilt sensors in current state-of-the-art mobile devices. Some early work by (Wilson and Pham, 2003; Wilson and Shafer, 2003) introduced the idea of an *XWand* which is a custom built pointing device that controls electronic devices in an intelligent environment using a variety of sensors in combination to support pointing and gesture recognition tasks. A museum guide is described in (Chan et al., 2005) to help visitors quickly locate exhibits using orientation aware handheld devices. (Baillie et al., 2005) reports on an interaction method where a user points a mobile device at a building to view a virtual representation of it at different times in the past. In (Essl and Rohs, 2007), their approach is to use the sensor data to turn the device into a musical instrument using shaking and sweeping gestures. The *Shoogle* project

(Williamson et al., 2007) aims to use inertial sensing to provide an eyes-free vibrotactile display that mimics objects rattling around inside the device. Such active use of sensors provides for a rich multimodal interaction that can be used without any visual attention.

The majority of current research focuses on providing enhanced navigation capabilities in the area of mobile spatial interaction. The *Point-to-Discover* GeoWand (Simon and Frohlich, 2007b; Simon and Frohlich, 2008) is a system and application framework for orientation-aware location-based mobile services. This application demonstrates how their custom-built device can be used as a pointing tool to display web-based information about bars and restaurants on the device. A very similar approach is taken by (Frank, 2009) with the *iPointer* application which is based on an augmented reality engine and a thin client api that provides a local mobile search based on GPS and an eCompass and delivers content, such as pictures, menus, and audio overviews, which are streamed back to the user's phone. A rather different approach is taken by (Robinson et al., 2008). Using their *Point-to-GeoBlog* application, users are able to select landmarks by using the point and tilt functionality of a custom built device. No content is provided up front but later when the user logs onto a computer with more visually adequate display capabilities.

A detailed usability study by (Robinson et al., 2008) reported that the most intuitive approach was to provide a simple point and tilt interface over a visual map and also a preference for remote tagging that allows users to select landmarks beyond their line-of-sight. These results confirmed a comparative outdoor study by (Frohlich et al., 2006) that tested conceptual designs for 4 interaction areas considered important for spatial information applications (SIA). In addition, the information *Pull* technique, where the user decides what information to view and has control over it (Persson et al., 2002; Strachan and Murray-Smith, 2009; Strachan et al., 2007) was the most intuitive and preferred data interaction approach by users. Whereas the *Push* technique, where all information is automatically presented to the user, is not easily managed on constrained mobile devices.

An alternative approach that combats the types of problems related to too much data being presented to the user is to restrict the search space based on certain criteria. In (Gardiner and Carswell, 2003), the approach is to restrict the search space to a users field-of-view using the concept of an observer’s 2-dimensional query frustum to determine what the user can actually see from their position in the environment. The use of the *Point-to-Discover* block model and visibility computation algorithm in (Simon and Frohlich, 2007b) uses a rather different approach to determine what buildings the observer can actually see from a single vantage point. In (Simon and Frohlich, 2007a), this idea is extended with the development of a local visibility model that introduces the concept of “billboards” as a mechanism to identify what buildings the user can see.

This type of egocentric visibility will be a primary focus of this paper. We consider only possibilities in relation to what a user can physically see from their current position by using visibility analysis to do so. Utilising this type of visibility shape on a mobile platform is a new concept and will be a key aspect of the research.

3 Visibility Analysis

There are a number of methods used in the current research described above for performing line-of-sight queries. This section gives an overview of the technologies and methods used by a number of them and describes some emerging possibilities.

3.1 Ray Tracing

The ray tracing process is a fundamental process in this area and works by simulating the light that travels within a space. In many applications, this process is performed backwards to determine the visibility from one or multiple points (e.g. user positions), which instead of perceiving the light emitted from the objects (e.g. buildings or other infrastructure), the ray is transmitted from a given point outwards in all directions. By retrieving all the intersections from a generated ray and the objects, a polygon in a 2D environment and a volume in a 3D environment can be constructed that represents the visibility from the point in 2D and 3D respectively (Figure 1). This approach is termed *Line-of-Sight* (LOS) in Geographic Information Systems (GIS).

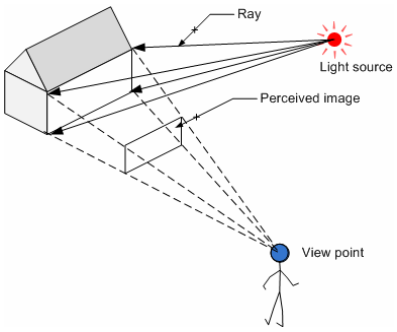


Fig. 1. Ray Tracing Process

3.2 Line-of-Sight

The *Line-of-Sight* function has been integrated in most commercial GIS software. It determines the visibility from the observer's position to a target point considering DTM fluctuations. The path from the viewpoint to the target point is one of the rays emitted from the viewpoint. In effect, some of the projected rays are truncated where they intersect with obstacles (e.g. building blocks, etc) while on route to the target. This collection of interaction points forms a convex polygon or volume representing the visibility area of a specific viewpoint. In a 2D environment, where the elevation of the objects is not taken into account, the line-of-sight function to establish the visibility of a particular point in space can be simplified by recording all the intersection points between the rays and obstacles surrounding it in the horizontal plane (Figure 2). However, in a 3 dimensional space, the ray is not just projected in

the horizontal plane but also in the vertical plane using the tilt angle from the horizontal plane that ranges from 0 to 360 degrees. This particular object is referred to as an *Isovist* in a 3D environment (Benedikt, 1979). An isovist is described as a visibility polygon comprised of the set of all points visible from a given vantage point in space with respect to an environment. Isovists were first introduced as a method for analysing space in space syntax research by (Benedikt, 1979). In (Jiang and Liu, 2009) this concept has been extended recently with the automatic generation of an axial map which can be used to generate an isovist. Fortunately, a similar function available in GIS is known as a *Viewshed* for geodetic survey applications.

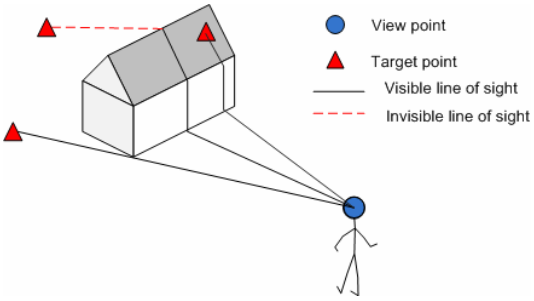


Fig. 2. Line-of-Sight Process

3.3 Viewshed Analysis

Viewshed analysis adopts the same approach as line-of-sight analysis. However, instead of examining the single path from the viewpoint to the target point, a beam of rays is generated from the viewpoint in the horizontal plane. Concurrently, a beam of rays is generated vertically along each ray path in the horizontal plane and the azimuth is taken into consideration within a certain range. For instance, in the example in Figure 3 the viewshed is based on the viewpoint and the range (based on the azimuth) combined with the horizontal angle or tilt, where the actual visible space is a volume excluding the section of the 3D object that intersects with the surface or building.

To estimate the visibility in all directions from the viewpoint using viewshed analysis, the azimuth in the horizontal plane and the view angle in the vertical plane can be extended to a full range, which generates the rays from a point between 0 and 360 degrees, forming a frustum shaped viewshed. One existing application of this idea is the *Threat Dome*, which is used for estimating all possible locations in space that are visible from a given point and is utilised mainly in military defence situations. (ESRI, 2009; Skyline, 2009).

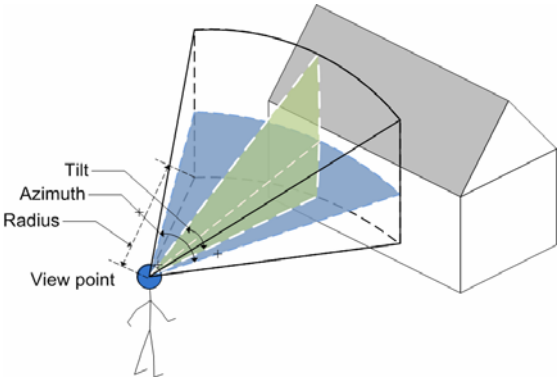


Fig. 3. Viewshed analysis in geodetic surveying

Adopting such an approach to measure the visible space within a certain radius can significantly reduce the number of calculations required to determine the possible intersection points. This strategy is similar to radar scanning the environment with a limited signal distance (strength). Essentially, it is an extreme case of the radial line-of-sight, which is derived by calculating the basic line-of-sight.

3.4 Threat Dome Analysis

In contrast to viewshed analysis, capturing the visible volume from a point taking into account all existing obstacles (objects) in the environment can be very computationally intensive. The threat dome approach specifies a radius defining the view distance from a point and hence the rays emitted from the viewpoint form a sphere. By determining the intersections of the rays with obstacles in different levels of elevation within the sphere, the actual visible space within this specified radius can be determined (Figure 4).

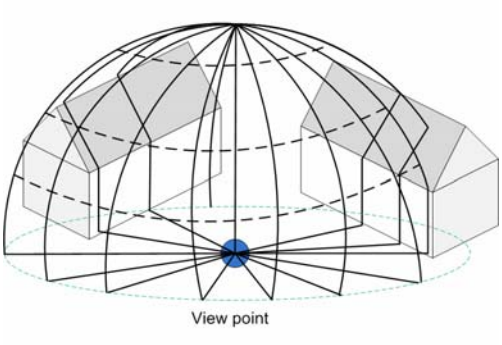


Fig. 4. Example of Threat Dome

4 TellMe

The *TellMe* application is the mobile application that requests information from two modules, namely the *LocateMe* and *DirectMe* modules and uses this information to perform directional queries using the *TellMe Server* against various spatial data sets. Two main approaches are investigated in relation to implementing the *TellMe Server*.

One approach is to host the spatial dataset on the mobile device itself. In this case the data retrieved from *LocateMe* and *DirectMe* is used to perform spatial queries locally on the mobile device. The results of the queries are displayed to the user by the *TellMe Mobile* application. Because of limited memory and the processing speeds required to perform spatial queries on large geographic data sets, this type of architecture was deemed unsuitable.

The alternative approach (i.e. thin client/server approach) is to request the required data from *LocateMe* and *DirectMe* and perform spatial queries on a dataset hosted on an external server. The parameters are passed to a web application, which in turn carries out the queries on a spatial database to determine any spatial interaction between the users location and orientation and the dataset. The results are subsequently displayed to the user in the web browser on the mobile device. This architecture is illustrated in Figure 5.

This approach of using a web browser to display the returned data follows current trends in this area. The argument for using web browsers for as many on-device functions as possible is because of the increasing complexity associated with developing applications to run on many different devices (W3C, 2008a). To avoid developing a different application for each operating system, a more attractive solution is to develop applications using a web scripting language capable of producing highly interactive applications that perform the same functions as desktop applications without the concern about underlying operating system functions and restrictions.

Taking this approach a step further, it is now becoming more acceptable to let the browser access more different types of data as well. Using a set of JavaScript APIs, it is even possible to access some of the hardware on a mobile device directly from the browser. For example, this is one of the newest features of *Google Gears* (Google, 2008) where the *Geolocation* API allows an application access to the GPS hardware on the device. Using this approach, if all sensor data was made available to a web application, it would be possible to eliminate the need for an on-device application altogether, essentially making the *TellMe* application entirely web-based.

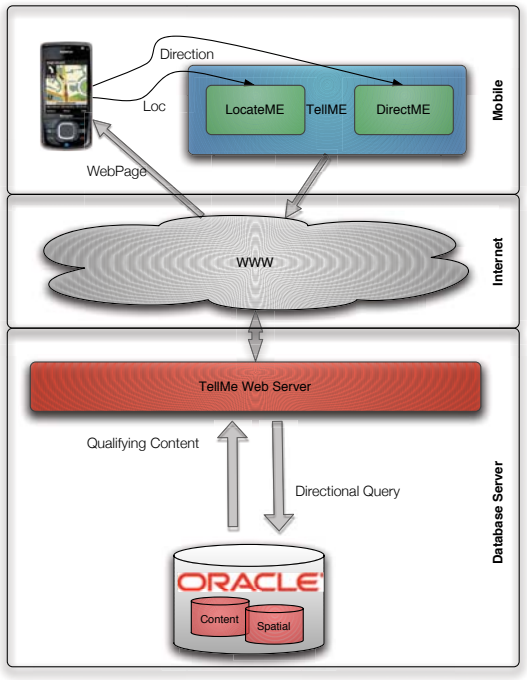


Fig. 5. Overall *TellMe* Architecture

4.1 LocateMe

The *LocateMe* module is used in conjunction with the *DirectMe* module to gather information about a user's current position and orientation. This data is then used to execute directional queries against various data sets both internal and external to the phone. The *LocateMe* module is based on a hybrid positioning system that utilises GSM, Wi-Fi and Bluetooth radio signals in addition to GPS to determine location. As this is not the focus of this paper, a more comprehensive description of this technology can be found in (Rooney et al., 2007b).

4.2 DirectMe

The *DirectMe* module is one of two modules that provide data to the *TellMe* application. Its function is to determine the direction that the mobile device is currently pointing by using a digital compass and tilt sensors. This data is then collected on request from the *TellMe* application and synchronised with the location data coming from the *LocateMe* module. The architecture of the *DirectMe* module is similar to the *LocateMe* module where each technology (i.e. - compass and tilt

sensors) has a native hardware spotter that relays data to a higher-level component synchronises data from each spotter. This architecture is illustrated in Figure 6.

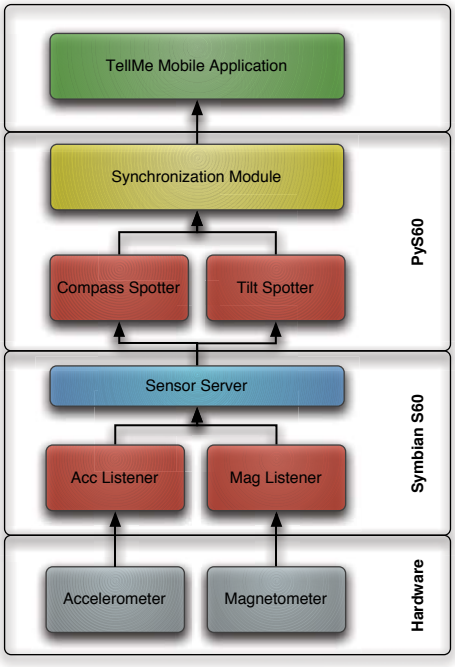


Fig. 6 DirectMe Design

This type of architecture is required in order to overcome some restrictions imposed in relation to access to various mobile device hardware components (e.g. - compass and tilt sensors) from particular APIs.

4.3 Mobile Device Hardware/Sensors

There are three main mobile platforms that currently provide mobile devices that contain the required sensor hardware for the *TellMe* system. The *Nokia 6210 Navigator* (Nokia, 2008) has a digital compass and tilt sensors and the API providing access to these sensors has just (winter '09) been back ported from Symbian S60 5th Edition to Symbian S60 3rd Edition FP2. Using this version of the API, the *DirectMe* module can ascertain the current heading and orientation of the device. However, there have been recent reports that the performance of these integrated sensors is of poor quality (Essl and Rohs, 2007). Another hardware option is the *HTC Dream* (a.k.a. *Google phone*), with integrated digital compass and accelerometers. This device runs the Android operating system from *Google* and is currently not available in Ireland (HTC, 2008). The most recent possibility in terms of hardware suitability is

the *Apple iPhone 3GS* (Apple, 2009). In addition to providing WiFi and accelerometer access there is now a compass available making it suitable also. With this limited number of options available in relation to acquiring orientation data from mobile devices integrated or "on-board" sensors, another option is to use external sensors packs such as the *SHAKE*. The *SHAKE SK6* is an external sensor pack with digital compass, accelerometer, and tactile feedback. The even smaller SK7 has been released in Q1 2009. This sensor pack communicates with a mobile device via Bluetooth. Unlike current cellphone sensors, the *SHAKE* device has better quality sensors and a number of filters on board to reduce any noise introduced by the mobile phones antenna. The *SHAKE SK7* is shown in Figure 7.

However, following a review of these options, it was decided that the *DirectMe* module would collect data from a number of MEMS (Micro Electro-Mechanical Systems) sensors on a *Nokia 6210 Navigator* (Figure 7) in favor of the SHAKE as these sensors are integrated. In particular, the sensors that we use are magnetometers, which are capable of sensing the magnetic field surrounding the device. Using the magnetic field to determine magnetic north, it is possible to calculate compass bearing (Essl and Rohs, 2007). Furthermore, accelerometers are used to measure acceleration or the rate of change of velocity with respect to time.

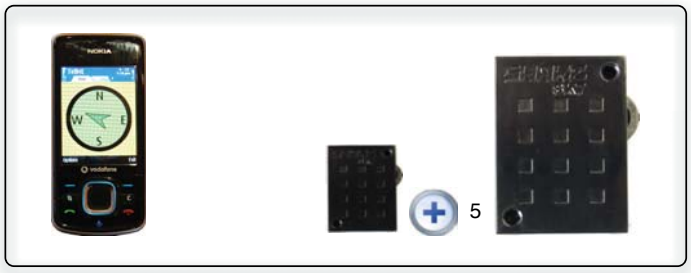


Fig. 7. Nokia 6210 Navigator and SHAKE SK7 sensor pack

4.4 TellMe Server

The TellMe server is essentially a spatial application server that is used to perform all the complex spatial queries in the system. It is responsible for communicating with the *TellMe Mobile* application, which collects data (including location, direction and orientation) from the *LocateMe* and *DirectMe* modules on the mobile device. This data is communicated wirelessly to the spatial application server and used to perform spatial queries against the *Oracle Spatial* 3D Database.

The *TellMe Server* is based on ESRIs *ArcGIS Server* platform and is used to perform the complex queries that are required to determine the mobile spatial interaction between the users line-of-sight and the 3d database. This platform provides server extensions for many of the traditional spatial query functions found in most GIS. Using these extensions, it is now possible to perform these types of spatial queries in a mobile context that were previously only possible in a desktop setting. Another important reason for this choice of server is based around the issue of

scalability; using this software ensures that the system is scalable and can manage a large number of users efficiently.

4.5 3D Database

To perform directional queries, a 2D spatial database can be used and is the minimum requirement in order to do so. These databases support extensive 2D feature types and indexing techniques for performing spatial queries. 2D spatial queries can be performed effectively with a standard spatial database in an efficient manner and can identify objects (i.e. buildings) that intersect with a direction vector for example. Some types of queries possible are illustrated in Figures 10, 11 & 12.

However, to perform 3D spatial queries this process becomes somewhat more complex. In comparison, a 3D spatial query should be able, for example, to not only identify what building a direction vector is intersecting with but also the floor of the building it is directed at in a 3D Euclidean space. This means a true 3D database should support three-dimensional data types such as point, line, surface and volume in its geometric data model, be capable of indexing the data and must also offer functions and operations embedded in its spatial query language that can operate on these data types (Schön et al., 2009). In fact, these requirements significantly reduce the number of options that are available to us in terms of being able to perform these types of queries. There are two main options here.

4.5.1 Oracle 11g

Beginning with the 11g version of Oracle it is now possible to store, index and query 3D objects using the *sdo_geometry* data type. Using this type, it is now possible to store point, line, polygon, polygon with hole and collection data types in 2D and 3D (Schön et al., 2009). An example of the types of data that can be stored is illustrated in Figure 8 (b).

4.5.2 ESRI ArcGIS

To support this rising trend in 3D data storage, ESRI developed a native volumetric geometry feature type called the multipatch feature supported by its geo-database models that is treated like any other geometry type in the database. The multipatch is constructed of triangle strips and fans and defines objects boundaries using triangular faces. This is shown in Figure 8(a)

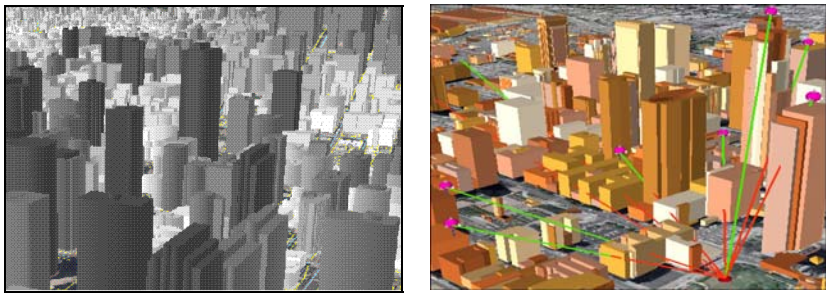


Fig. 8. (a) ArcGIS Multipatch 3d objects (b) Oracle SDO_Geometry

In our case, we use Oracle 11g and its exclusively 3D forms: simple solid, composite surface, composite solid and collection to represent our data in the database. The data is comprised of 3d data that is based on Ordinance Survey Ireland's (OSI, 2009) 2d vector data that has extruded using height values from airborne LiDAR scans. The result is a block model of the NUI campus and Dublin city centre. There are also some detailed building furniture models of the NUI campus buildings, illustrated in Figure 8b. This data is spatially indexed and is queried using a 3D query window generated by the data collected from the mobile device sensors. All attribute data (building name, class timetables, etc.) presented to the user is also stored in the Oracle database. To perform 3D queries an extended set of operators can be used as support for 3D data is restricted to the SDO_Filter, SDO_Anyinteract, SDO_Within_Distance, and SDO_NN (nearest neighbour) operators using the Geographic-3D coordinate system. An example of a typical 3D query or frustum query is shown in Figure 9.

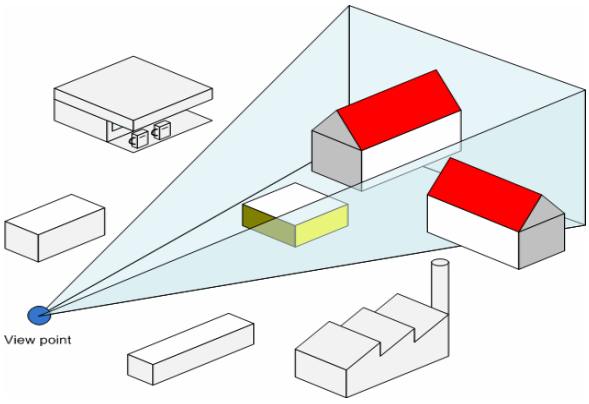


Fig. 9. Spatial query using view frustum

5 Ego-Visibility

In this section, the different types of queries that are performed by the *TellMe* application are discussed, all of which are based on the users egocentric point-of-view. Egocentric visibility refers to the portion of a search space that is visible to a user at a particular time based on their location, direction and orientation. For example, in the case of the *TellMe* application, the user's visible query space acts as a secondary filter on data that is returned by the query processor by restricting it to contain only objects that are in the users field-of-view (FOV). The FOV therefore excludes the portions of the dataset obscured by buildings. This method is primarily used to identify points-of-interest (POI) other than buildings that are in the users FOV within a predefined distance from the user and is illustrated in Figure 11. This method can be used to identify objects in the distance that may be too small to point at directly but are still in the users FOV nonetheless, like a monument or statue. The algorithm to determine the searchable space in this instance builds on previous work outlined in (Gardiner and Carswell, 2003) where a 2D directional query processor was developed and used in a virtual environment for similar purposes. The following sections outline the queries that can be performed using the *TellMe* system.

5.1 Directional Query

Directional querying in terms of Mobile Spatial Interaction (MSI) is a method by which a device's position and orientation along 2-axis (horizontal and vertical) can be determined with the use of GPS and compass/tilt sensors. This data is then used to build a "query space" in the database to identify what object(s) the device is pointing at - in our case a campus building and any relevant information about it (Figure 10). When the user points a device at a building (creating a query vector), the device is able to identify what it is pointing at by determining the interaction between the query vector and the building data model stored in the database. The returned information can be communicated back to the user using visual, auditory or tactile interfaces.

Fig. 10. Identifying buildings using directional query vectors

This initial scenario is the simplest possible example of what we aimed to achieve in our research and it assumes that a users LOS is a straight-line vector. As this is not the case, we investigated other possibilities in terms of what the user can actually see. We look at ways of representing 2D and 3D query frustums that interact with objects based on LOS (i.e. direction) and visible open space (i.e. everything a user can see from a single point in all directions) or Egocentric Visibility (EgoViz).

5.2 Field-of-View Query

The field-of-view query maintains the use of the direction vector in the query process by producing a query frustum that closely represents a users actual field-of view. In contrast to the directional query, performing this type of query returns a list of results that identify not only what the user is pointing at but also everything in the users field-of-view (Figure 11). It is also possible to pre-select the layers that the user requires information about. For example, a user may only want information about the buildings that are in their field-of-view or alternatively they may want information about points-of-interest only. This categorisation of the required data helps to speed up the query process by eliminating sets of data not required.

Taking this concept a step further, the idea of the visibility polygon or Isovist is introduced. A visibility polygon is the portion of open space a user can see in all directions. Being able to determine a users visibility polygon in real-time as they navigate through the campus or city streets enables the *TellMe* system to deliver a much richer experience in terms of the relevance of the data to the user and the speed at which the data is delivered. As the visibility polygon identifies the area the user can actually see, this generates a much smaller search space reducing the size and complexity of the spatial queries to be performed.

Fig. 11. Identifying POI objects in a user's field-of-view

5.3 Isovist Query

Recent work in this area by (Simon and Frohlich, 2007a) describes the local visibility model (Lvis) that uses the concept of billboards to determine what buildings are in the users FOV in a 2.5D environment. To achieve this in our case, a different approach is taken based on work carried out by (Jiang and Liu, 2009) into the concept of Isovists and medial axes. Using Isovists, we attempt to automatically generate a panoramic (360º) line-of-sight search space for performing spatial queries on. This method should prove to be very effective as the fundamental aim is firstly to determine precisely what area in geographical terms is visible to the user in all directions and secondly to determine what objects inside this area is of interest to the user. This idea is illustrated in Figure 12.

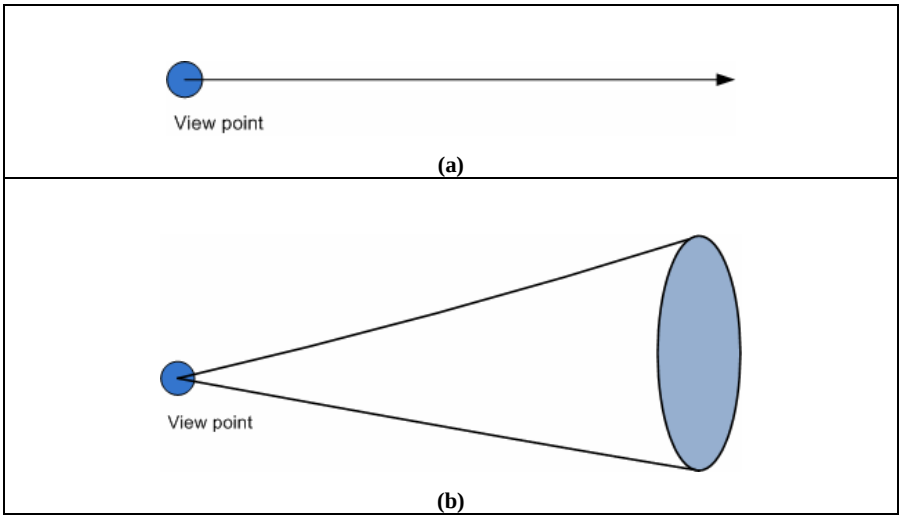
Fig. 12. Egocentric Visibility using IsovistExplorer 360°

5.4 Frustum Query

Extending the 2D Isovist to work in 3D space is a primary objective of this work. In the case of the frustum query, information about relevant artifacts that lie within the sections of open space (i.e. the space between the buildings in all directions up/down/sideways) that has been identified as visible to the user are queried and returned.

Fig. 13 3D Visibility Query

In this case, the second generation of the 3D Isovist is represented by a 3D object that represents a users actual LOS omitting the geometry of any other objects that interact with it (illustrated in Figure 13). Building this type of 3D Isovist query requires an extensive 3D database of buildings and building furniture at floor and room detail in order to utilise the query frustum geometry. Examples of the types of geometries used are illustrated in Figure 14.



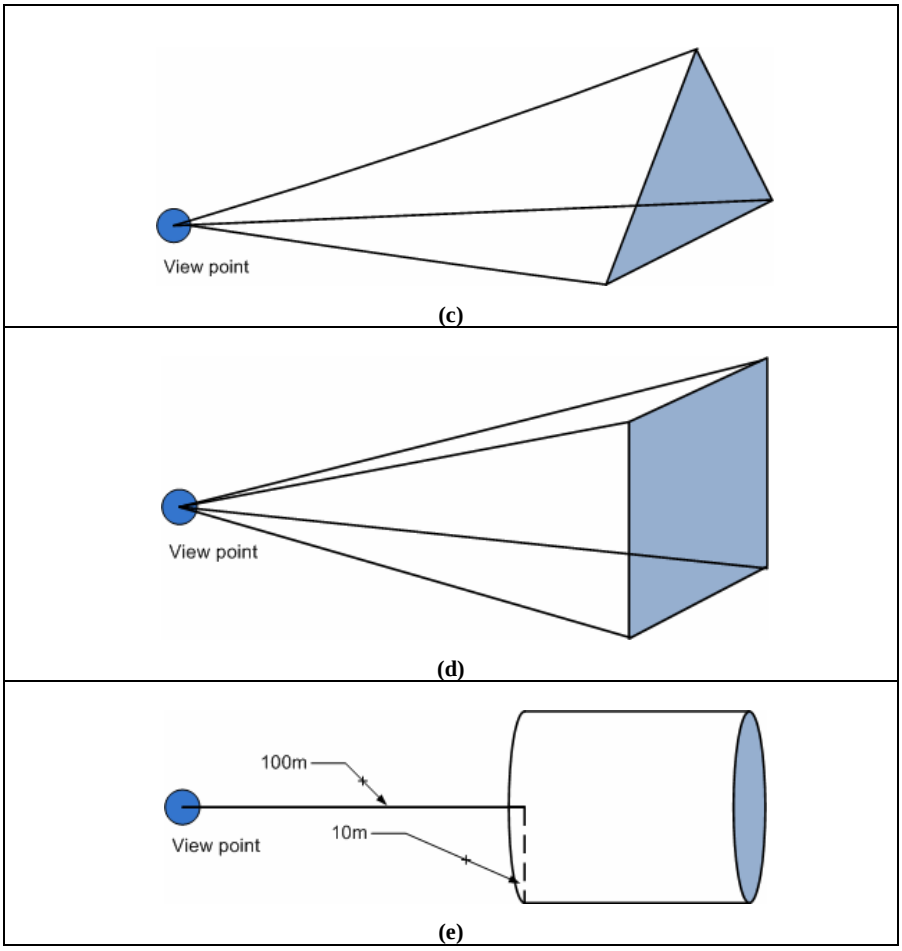


Fig. 14. Examples of 3D Visibility Queries

5.5 Ego-Dome

The Ego-Dome is essentially an extension of the 2D Isovist described above except the visible open space of the user from a particular location is reproduced in 3D. This concept of querying all open space in this manner has been explored previously but usually only using standalone desktop applications to do so, offline. In the case of the *TellMe* system, the *Ego-Dome* is created using the current location of the user (Figure 15). Any objects that interact with the dome are detected and the required results are presented to the user using visual, auditory or tactile interfaces. This type of spatial interaction is very useful for a host of applications in terms of real-time feedback that

gives the user the ability to interact and have knowledge about an environment in realtime.

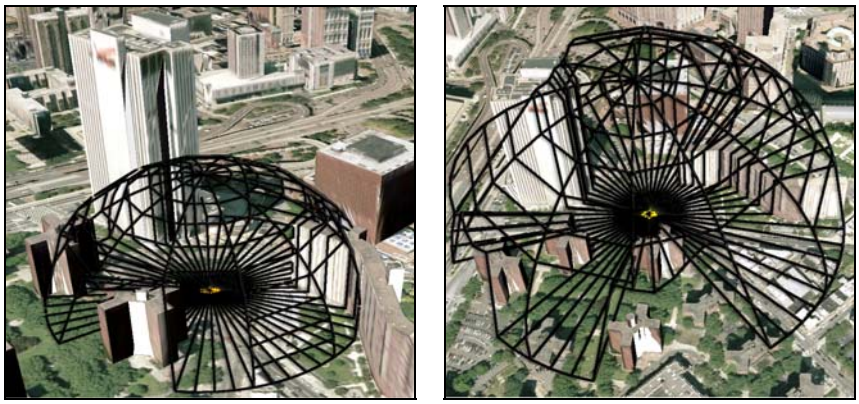


Fig. 15. Ego-dome being used to determine the visible space within a certain radius

6 Conclusions

This mobile based spatial interaction system (Egoviz), which is based on the *TellMe* application is designed to allow users to interact in a context sensitive way with information using current mobile phone technology. Current devices are beginning to offer hardware such as digital compasses and tilt sensors. With the use of these technologies, a wide spectrum of applications are emerging. The development of the *DirectMe* module and its synchronisation with the *LocateMe* module is ongoing. Concurrently, the development of the complete *TellMe* mobile application is taking place and provides a framework for testing our directional query techniques.

In relation to performing directional queries, there are a number of fundamental issues that have been highlighted in terms of the quality of the data that is retrieved from the sensors. In (Simon and Frohlich, 2008) the quality of the sensor data from a custom built sensor pack is analysed and compared to actual data showing good results for performing directional queries. With the quality of the sensor data being a pivotal aspect of directional queries in terms of the relevance of the data returned and the move towards making this type of interaction available on mainstream mobile devices, we intend to carry out two important sets of tests. Firstly we investigated the quality of selected external sensors (i.e. SHAKE and Nokia LD-4W GPS) and secondly we will compare these results with the quality of the data from the integrated sensors in the "off-the-shelf" phone (i.e. Nokia 6210 Navigator).

Regarding egocentric visibility, research will be carried out to help us better identify what comprises exactly a user's FOV. Initially we have studied work by (Jiang and Liu, 2009) in the area of spatial planning and in particular Isovists by developing a 2D EgoVis filter to determine a users true LoS in 360°. In due course, the technical feasibility of extending to a true 3D directional query processor will be investigated using the idea of the threat dome as an example.

References

Baillie, L., Kunczier, H. and Anegg, H., 2005. Rolling, Rotating and Imagining in a Virtual Mobile World, 7th international conference on Human computer interaction with mobile devices & services, Salzburg, Austria

Benedikt, M.L., 1979. To take hold of space: isovists and isovist fields. Environment and Planning B 6(1): 47 – 65

Chan, L.-W., Hsu, Y.-Y., Hung, Y.-P. and Hsu, J.Y.-J., 2005. Orientation-Aware Handhelds for Panorama-Based Museum Guiding System, UbiComp 2005 Workshop: Smart Environments and thier Applications to Cultural Heritage.

ESRI, 2009. ESRI Globe.

Essl, G. and Rohs, M., 2007. ShaMus - A Sensor-Based Integrated Mobile Phone Instrument, Proceedings of the International Computer Music Conference (ICMC), Copenhagen.

Frank, C., 2009. Intelliogent Spatial Technologies, Maine.

Frohlich, P., Simon, R., Baillie, L. and Anegg, H., 2006. Comparing Conceptual Designs for Mobile Access to Geo-Spatial Information, 8th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 06), Helsinki,Finland.

Gardiner, K. and Carswell, J., 2003. Viewer-Based Directional Querying for Mobile Applications, Third International Workshop on Web and Wireless Geographical Information Systems W2GIS, Rome, Italy.

Google, 2008. Google Gears API.

HTC, 2008. HTC - Dream.

Jiang, B. and Liu, X., 2009. AxialGen: a research prototype for automatically generating the axial map, Submitted to 11th International Conference on Computers in Urban Planning and Urban Mnagement, Hong Kong.

Kilfeather, E., Carswell, J., Gardiner, K. and Rooney, S., 2007. Urban Location Based Services using Mobile Clients: The ICiNG Approach, GISRUK, Maynooth, Ireland.

Nokia, 2008. Sense Your Location.

OSI, 2009. Osi Website.

Persson, P., Espinoza, F., Fagerberg, P., Sandin, A. and Cöster, R., 2002. GeoNotes: A Location-based Information System for Public Spaces. Readings in Social Navigation of Information Space: 151-173.

Robinson, S., Eslambolchilar, P. and Jones, M., 2008. Point-to-GeoBlog: Gestures and Sensors to Support User

Generated Content Creation, 10th international conference on Human computer interaction with mobile devices and services, Amsterdam, The Netherlands.

Rooney, S., Gardiner, K. and Carswell, J., 2007a. AN OPEN SOURCE APPROACH TO WIRELESS POSITIONING TECHNIQUES, The 5th International Symposium on Mobile Mapping Technology (MMT,07), Padua, Italy.

Rooney, S., Gardiner, K. and Carswell, J., 2007b. Wireless Positioning Techniques – A Developers Update, Web and Wireless Geographical Information Systems, pp. 162-174.

Schön, B., Laefer, D.F., Morrish, S.W. and Bertolotto, M., 2009. Three-Dimensional Spatial Information Systems: State of the Art Review. Recent Patents on Computer Science, 2: 21-31.

Simon, R. and Frohlich, P., 2007a. A Mobile Application Framework for the Geospatial Web, Proceedings of the 16th international conference on World Wide Web, Banff, Alberta, Canada.

Simon, R. and Frohlich, P., 2007b. The Point to Discover GeoWand, 9th International Conference on Ubiquitous Computing (UbiComp 07), Innsbruck, Austria.

Simon, R. and Frohlich, P., 2008. GeoPointing: Evaluating the Performance of Orientation-Aware Location-Based Interaction Under Real-World Conditions, 4th International Conference on LBS and TeleCartography Taylor & Francis, Hong Kong.

Skyline, 2009. Skyline Terraserver.
Strachan, S. and Murray-Smith, R., 2009. Bearing-based selection in mobile spatial interaction. Personal and Ubiquitous Computing, 13(4).
Strachan, S., Williamson, J. and Murray-Smith, R., 2007. Show me the way to monte carlo: density-based trajectory navigation, Proceedings of the FP 205 SIGCHI conference on Human factors in computing systems, New York, USA, pp. 1245-1248
W3C, 2008a. Mobile Web Application Best Practices.
W3C, 2008b. Mobile Web Best Practices 1.0.
Williamson, J., Murray-Smith, R. and Hughes, S., 2007. Shoogle: Excitatory Multimodal Interaction on Mobile Devices, Proceedings of ACM SIG CHI Conference, San Jose.
Wilson, A. and Pham, H., 2003. Pointing in Intelligent Environments with the WorldCursor, Interact.
Wilson, A. and Shafer, S., 2003. XWand: UI for Intelligent Spaces, CHI, Fort Lauderdale, Florida.